GDML

- Importing and exporting detector descriptions
GDML components

- GDML (Geometry Description Markup Language) is defined through XML Schema (XSD)
  - XSD = XML based alternative to Document Type Definition (DTD)
  - defines document structure and the list of legal elements
  - XSD are in XML -> they are extensible

- GDML can be written by hand or generated automatically in Geant4
  - 'GDML writer' allows exporting a GDML file

- GDML needs a “reader”, integrated in Geant4
  - 'GDML reader' imports and creates 'in-memory' the representation of the geometry description
GDML – Geant4 binding

• XML schema available from [http://cern.ch/gdml](http://cern.ch/gdml)
  – Also available within Geant4 distribution
    • See in `geant4/source/persistency/gdml/schema/`
    – Latest schema release GDML_3_0_0 (as from 9.2 release)
  • Requires XercesC++ XML parser
    – Available from: [http://xerces.apache.org/xerces-c](http://xerces.apache.org/xerces-c)
    – Tested with versions 2.8.0 and 3.0.1
• Optional package to be linked against during build
  – `G4LIB_BUILD_GDML` and `XERCESCROOT` variables
  – Examples available: `geant4/examples/extended/persistency/gdml`
CMS detector through GDML

~19000 physical volumes

Geant4 CMS geometry imported in Root through GDML
LHCb detector through GDML

~5000 physical volumes
Using GDML in Geant4

to write:

```cpp
#include "G4GDMLParser.hh"
G4GDMLParser parser;
parser.Write("g4test.gdml", pWorld, true, "path_to_schema/gdml.xsd");
```

`instantiate GDML parser`

`Concatenate or not pointers to entity names`

`pass the 'top' volume to the writer`

`Activate or de-activate schema validation`


to read:

```cpp
parser.Read("g4test.gdml", true);
```

`get pointer to 'top' world volume`

```cpp
pWorld = GDMLProcessor::GetInstance()->GetWorldVolume();
```
Any geometry tree can be dumped to file
  - ... just provide its physical volume pointer (pVol):
    `parser.Write("g4test.gdml", pVol);`

A geometry setup can be split in modules
  - ... starting from a geometry tree specified by a physical volume:
    `parser.AddModule(pVol);`
  - ... indicating the depth from which starting to modularize:
    `parser.AddModule(depth);`

Provides facility for importing CAD geometries generated through STEP-Tools

Allows for easy extensions of the GDML schema and treatment of auxiliary information associated to volumes

Full coverage of materials, solids, volumes and simple language constructs (variables, loops, etc...)

Using GDML in Geant4 - 2
Importing CAD geometries with GDML

- CAD geometries generated through STEP-Tools (stFile.geom, stFile.tree files) can be imported through the GDML reader:
  
  ```
  parser.ParseST("stFile", WorldMaterial, GeomMaterial);
  ```

- Tools like FastRad allow for importing CAD STEP files and directly convert to GDML
GDML processing performance

• GDML reader/writer tested on
  – complete LHCb and CMS geometries
  – parts of ATLAS geometry
    • full ATLAS geometry includes custom solids
• for LHCb geometry (~5000 logical volumes)
  – writing out ~10 seconds (on P4 2.4GHz)
  – reading in ~ 5 seconds
  – file size ~2.7 Mb (~40k lines)
• for CMS geometry (~19000 logical volumes)
  – writing out ~30 seconds
  – reading in ~15 seconds
  – file size ~7.9 Mb (~120k lines)
GDML as primary geometry source

- Linear Collider
  - Linear Collider Detector Description (LCDD) extends GDML with Geant4-specific information (sensitive detectors, physics cuts, etc)
  - GDML/LCDD is generic and flexible
    - several different full detector design concepts, including SiD, GLD, and LDC, where simulated using the same application
GDML as primary geometry source - 2

- Space Research @ ESA
  - Geant4 geometry models
    - component degradation studies (JWST, ConeXpress,...)
    - GRAS (Geant4 Radiation Analysis for Space)
  - enables flexible geometry configuration and changes
  - main candidate for CAD to Geant4 exchange format

ConeXpress
• Anthropomorphic Phantom
  – Modeling of the human body and anatomy for radioprotection studies
  – no hard-coded geometry, flexible configuration
Exercise 1c

• GDML